Decision Tree | Assignment

Question 1: What is a Decision Tree, and how does it work in the context of classification?

-> A **Decision Tree** is a supervised machine learning algorithm that is widely used for **classification** and **regression** tasks. In the context of **classification**, it works by learning a series of decision rules from the input features that split the data into classes.

A Decision Tree is a **flowchart-like structure**:

- **Nodes** represent features (attributes).
- **Branches** represent decision rules based on feature values.
- **Leaves** represent class labels (in classification) or values (in regression).

It recursively splits the data into subsets based on the value of input features, aiming to create groups that are as homogeneous as possible with respect to the target variable.

Example

Imagine a dataset to classify whether someone buys a computer:

Age	Income	Student	Buys Computer
<=30	High	No	No
>40	Medium	Yes	Yes
31–40	Low	Yes	Yes

Question 2: Explain the concepts of Gini Impurity and Entropy as impurity measures. How do they impact the splits in a Decision Tree?

-> Gini Impurity measures the **probability of incorrectly classifying** a randomly chosen element if it were labeled according to the distribution of labels in the node.

Formula:

For a node with *KK*K classes:

*Gini=1−∑i=1Kpi2Gini = 1 - \sum_{i=1}^{K} p_i^2*Gini=1−i=1∑K pi2

Where:

- *pip_i*pi = proportion of examples belonging to class *ii*i in the node

Interpretation:

- Gini = 0: Node is **pure** (all samples belong to one class).
- Higher Gini → More **mixed** or impure node.

Example:

If a node has 70% of Class A and 30% of Class B:

*Gini=1−(0.72+0.32)=1−(0.49+0.09)=0.42Gini = 1 - (0.7^2 + 0.3^2) = 1 - (0.49 + 0.09) = 0.42*Gini=1−(0.72+0.32)=1−(0.49+0.09)=0.42

Entropy measures the **amount of uncertainty or disorder** in a set. Used in the ID3 algorithm, it comes from **information theory**.

Formula:

*Entropy=−∑i=1Kpilog2(pi)Entropy = -\sum_{i=1}^{K} p_i \log_2(p_i)*Entropy=−i=1∑K pi log2 (pi)

Where *pip_i*pi is the same as above.

Interpretation:

- Entropy = 0: All data points belong to one class (pure).
- Maximum entropy occurs when classes are evenly split (maximum uncertainty).

Example:

Same as above: 70% A, 30% B:

*Entropy=−(0.7log20.7+0.3log20.3)≈−(0.7×−0.514)−(0.3×−1.737)≈0.881Entropy = -(0.7 \log_2 0.7 + 0.3 \log_2 0.3) \approx - (0.7 \times -0.514) - (0.3 \times -1.737) \approx 0.881*Entropy=−(0.7log2 0.7+0.3log2 0.3)≈−(0.7×−0.514)−(0.3×−1.737)≈0.881

Aspect	Gini Impurity	Entropy (Information Gain)
Speed	Faster (no logs)	Slower (uses logs)
Sensitivity	Slightly more sensitive to class imbalance	More theoretically grounded
Preferred in	CART algorithm	ID3, C4.5 algorithms

Question 3: What is the difference between Pre-Pruning and Post-Pruning in Decision Trees? Give one practical advantage of using each.

-> Pre-Pruning (Early Stopping)

📌 Definition:

Pre-pruning **stops the tree from growing** once a certain condition is met **before** it fully develops.

✅ How it works:

It limits tree growth based on criteria such as:

- Maximum depth (`max_depth`)
- Minimum number of samples to split (`min_samples_split`)
- Minimum information gain threshold

▨ Practical Advantage:

Faster training time — since the tree stops early, it's more efficient and better suited for large datasets or real-time applications.

✂ Post-Pruning (Pruning After Growth)

📌 Definition:

Post-pruning **trims branches** from a **fully grown** tree that do not contribute significantly to prediction accuracy.

✅ How it works:

- The full tree is built.
- Then branches are removed if they don't reduce validation error significantly.
- Techniques: Reduced error pruning, cost-complexity pruning (used in CART).

Practical Advantage:

Better generalization — since pruning is based on validation performance, it typically leads to simpler, more robust models that perform better on unseen data.

Summary Table:

Feature	Pre-Pruning	Post-Pruning
When applied	During tree construction	After tree is fully built
Control parameters	`max_depth`, `min_samples_split`	Validation error, cost-complexity
Main benefit	Faster training, less memory	Improved accuracy on test data
Risk	Might stop too early (underfit)	Computationally expensive

Question 4: What is Information Gain in Decision Trees, and why is it important for choosing the best split?

-> **What is Information Gain?**

Information Gain measures the **reduction in entropy (uncertainty)** after a dataset is split on a particular feature.

In other words, it tells us **how much "information" a feature gives us** about the target variable.

Formula:

*Information Gain (IG)=Entropy(Parent)−∑k=1n∣Dk∣∣D∣⋅Entropy(Dk)\text{Information Gain (IG)} = \text{Entropy}(Parent) - \sum_{k=1}^{n} \frac{|D_k|}{|D|} \cdot \text{Entropy}(D_k)*Information Gain (IG)=Entropy(Parent)−k=1∑n ∣D∣∣Dk ∣ ⋅Entropy(Dk)

Where:

- *DD*D = original dataset
- *DkD_k*Dk = subsets after split

- *|Dk||D|\frac{|D_k|}{|D|}*|D||Dk | = weight of each subset
- Entropy = measure of class disorder

🔍 Why is Information Gain Important?

- It helps the decision tree **select the feature that best separates the classes**.
- A higher Information Gain means the split **produces more "pure" child nodes** (i.e., more class-homogeneous).
- Choosing the feature with the highest IG at each step **reduces the depth and complexity of the tree** while maintaining or improving classification accuracy.

🧠 Example:

Suppose we want to classify whether someone buys a product.

Before any split:

- 5 yes, 5 no → Entropy = 1.0 (maximum disorder)

If splitting on "Age" results in:

- Group 1: 4 yes, 1 no → Entropy = 0.72
- Group 2: 1 yes, 4 no → Entropy = 0.72

Then:

*IG=1.0−(510·0.72+510·0.72)=1.0−0.72=0.28IG = 1.0 - \left(\frac{5}{10} \cdot 0.72 + \frac{5}{10} \cdot 0.72 \right) = 1.0 - 0.72 = 0.28*IG=1.0−(105 ·0.72+105 ·0.72)=1.0−0.72=0.28

The higher the IG, the more effective the feature is at separating the data.

✅ Summary:

Concept	Description
Information Gain	Reduction in entropy after a split

Purpose	Helps choose the best feature to split on
Higher IG means	Better, more informative splits

Question 5: What are some common real-world applications of Decision Trees, and what are their main advantages and limitations?

-> Real-World Applications of Decision Trees

1. Healthcare

- **Use**: Diagnosing diseases based on symptoms, lab results, and patient history.
- **Example**: Predicting if a patient has diabetes or heart disease.

2. Finance

- **Use**: Credit scoring, fraud detection, loan approval.
- **Example**: Approving loan applications based on income, credit score, and employment history.

3. Marketing

- **Use**: Customer segmentation, churn prediction, product recommendation.
- **Example**: Targeting promotional offers based on purchasing behavior.

4. Retail and E-Commerce

- **Use**: Predicting sales, managing inventory, customer retention.
- **Example**: Recommending products based on purchase history.

5. Manufacturing

- **Use**: Quality control, predictive maintenance.
- **Example**: Predicting machine failures based on sensor data.

6. Education

- **Use**: Student performance prediction, dropout risk analysis.
- **Example**: Identifying students at risk of failing a course.

☑ Advantages of Decision Trees

Advantage	Description
Easy to understand	Visual and intuitive — like flowcharts, easy for non-experts to interpret.
Handles different data types	Works with both categorical and numerical data.
No feature scaling needed	No need for normalization or standardization.
Can model non-linear patterns	Captures complex interactions between features.
Fast and inexpensive	Computationally efficient to train and predict.

⚠ Limitations of Decision Trees

Limitation	Description
Overfitting	Deep trees can model noise, hurting generalization.
Unstable to small changes	Small changes in data can lead to a completely different tree structure.
Biased splits	Can favor features with more levels (many unique values).
Lower accuracy (alone)	Often outperformed by ensemble methods (e.g., Random Forest, XGBoost).

Pro Tip:

To overcome many limitations of decision trees, they are often used as **base learners** in ensemble methods like:

- **Random Forests**
- **Gradient Boosting (e.g., XGBoost, LightGBM)**

These ensembles increase accuracy and robustness.

Question 6: Write a Python program to: ● Load the Iris Dataset ● Train a Decision Tree Classifier using the Gini criterion ● Print the model's accuracy and feature importances (Include your Python code and output in the code box below.)

-> https://colab.research.google.com/drive/1Xgb3ZHGaKO0N-_Oj1Ak2BGre_cbKm5zb?usp=sharing

Question 7: Write a Python program to: ● Load the Iris Dataset ● Train a Decision Tree Classifier with max_depth=3 and compare its accuracy to a fully-grown tree. (Include your Python code and output in the code box below.)

-> Python program

Question 8: Write a Python program to: ● Load the Boston Housing Dataset ● Train a Decision Tree Regressor ● Print the Mean Squared Error (MSE) and feature importances (Include your Python code and output in the code box below.)

-> python program

Question 9: Write a Python program to: ● Load the Iris Dataset ● Tune the Decision Tree's max_depth and min_samples_split using GridSearchCV ● Print the best parameters and the resulting model accuracy (Include your Python code and output in the code box below.)

->python program

Question 10: Imagine you're working as a data scientist for a healthcare company that wants to predict whether a patient has a certain disease. You have a large dataset with mixed data types and some missing values. Explain the step-by-step process you would follow to: ● Handle the missing values ● Encode the categorical features ● Train a Decision Tree model ● Tune its hyperparameters ● Evaluate its performance And describe what business value this model could provide in the real-world setting.

-> python program